FxPlug in Depth

Advanced Topics

Session 308

These are confidential sessions—please refrain from filming or taking pictures.

Apple Applications

FxPlug in Depth

Advanced Topics

Dave Howell
FxPlug Engineering Manager

http://developer.apple.com/wwdc2005/

What You'll Learn

Advanced FxPlug Topics

- Retiming
- Optional Parameter Types
- Custom Parameter Types
- Custom Parameter UI
- On-screen Controls
- OpenGL Arcana

Retiming

Bending the Law of Unidirectional Causality

- For filters only (not generators)
- Use `FxTemporalImageAPI` Host API methods
- Request input image at specified time
  - Filtered or unfiltered
  - As bitmap or texture

Retiming

Getting an Unfiltered Bitmap

```
FxBitmap *image;
id api = [_apiManager apiForProtocol:@protocol(FxTemporalImageAPI)];

[api getSourceBitmap:&image withInfo:renderInfo atTime:t];
```

Retiming

Getting a Filtered Bitmap

```
FxBitmap *image;
id api = [_apiManager apiForProtocol:@protocol(FxTemporalImageAPI)];

[api getInputBitmap:&image withInfo:renderInfo atTime:t];
```

Retiming

Getting an Unfiltered Texture

```
FxTexture *image;
id api = [_apiManager apiForProtocol:@protocol(FxTemporalImageAPI)];

[api getSourceTexture:&image withInfo:renderInfo atTime:t];
```

Retiming

Getting a Filtered Texture

```
FxTexture *image;
id api = [_apiManager apiForProtocol:@protocol(FxTemporalImageAPI)];

[api getInputTexture:&image withInfo:renderInfo atTime:t];
```

Demo

Retiming

Pete Warden
Graphics Coder

http://developer.apple.com/wwdc2005/

Optional Parameter Types

Histograms and Gradients

Dave Howell

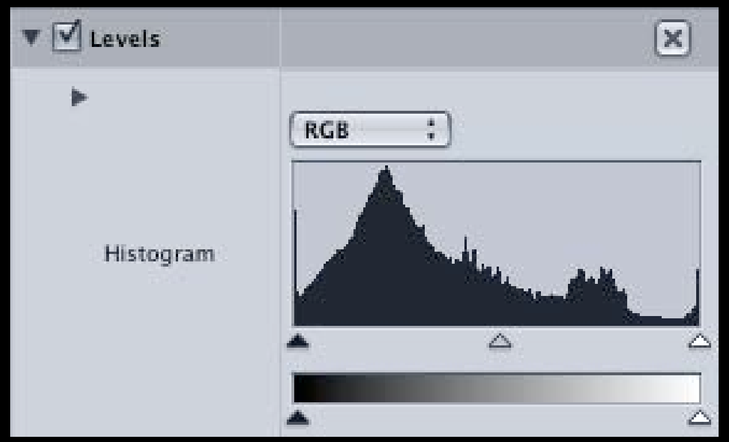
http://developer.apple.com/wwdc2005/

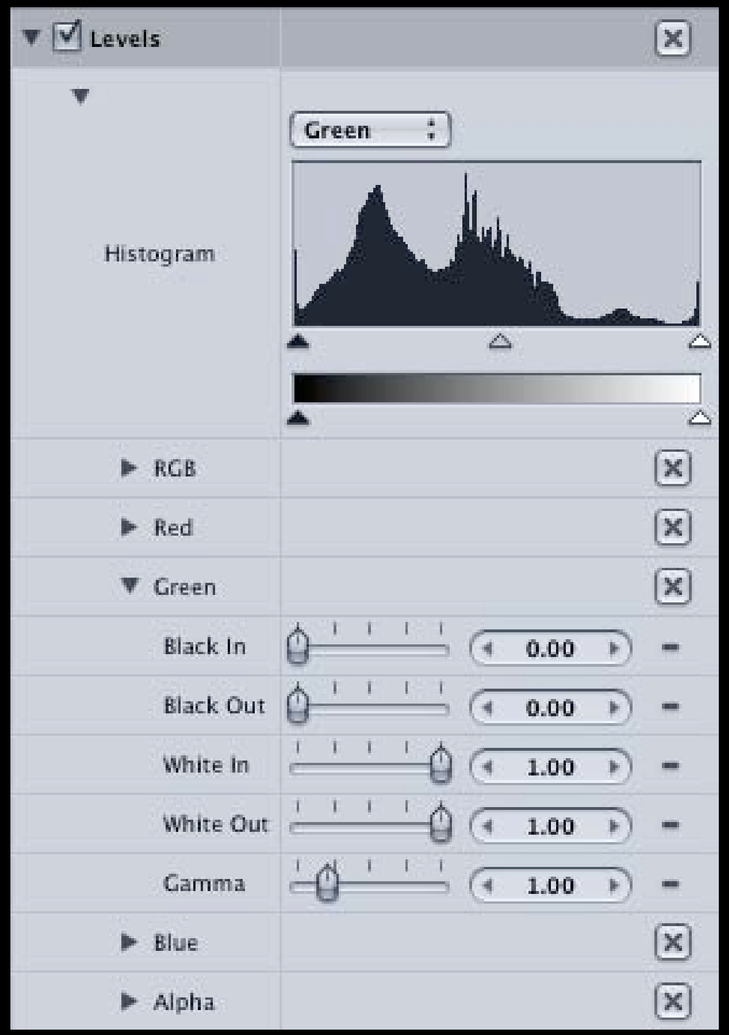
Optional Parameter Types

- May be unsupported by some host apps
- Use protocols in FxOptionalParameter.h
  - `FxOptionalParameterCreationAPI`
  - `FxOptionalParameterRetrievalAPI`
  - `FxOptionalParameterSettingAPI`
- Types
  - Histogram
  - Gradient

Special Parameter Types

Histogram





Special Parameter Types

Histogram Creation

```
id creator = [_apiManager apiForProtocol:
                    @protocol(FxOptionalParameterCreationAPI)];

[creator addHistogramWithName:@“Histogram”
                        parmId:kMyHistogramParamID
                    parmFlags:kFxParameterFlag_DONT_SAVE |
                              kFxParameterFlag_DONT_DISPLAY_IN_DASHBOARD];
```

Special Parameter Types

Histogram Value Retrieval

```
id getter = [_apiManager apiForProtocol:
                  @protocol(FxOptionalParameterRetrievalAPI)];
double blackIn, blackOut, whiteIn, whiteOut, gamma;

[getter getHistogramBlackIn:&blackIn
                   BlackOut:&blackOut
                    WhiteIn:&whiteIn
                   WhiteOut:&whiteOut
                      Gamma:&gamma
                 forChannel:channelNum
                   fromParm:kMyHistogramParamID
                     atTime:renderInfo.frame];
```

Special Parameter Types

Histogram Value Retrieval

```
id getter = [_apiManager apiForProtocol:
                  @protocol(FxOptionalParameterRetrievalAPI)];
double blackIn, blackOut, whiteIn, whiteOut, gamma;

[getter getHistogramBlackIn:&blackIn
                   BlackOut:&blackOut
                    WhiteIn:&whiteIn
                   WhiteOut:&whiteOut
                      Gamma:&gamma
                forChannel:channelNum
                   fromParm:kMyHistogramParamID
                     atTime:renderInfo.frame];
```

Channel Number
0 = RGB
1 = red
2 = green
3 = blue
4 = alpha

Special Parameter Types

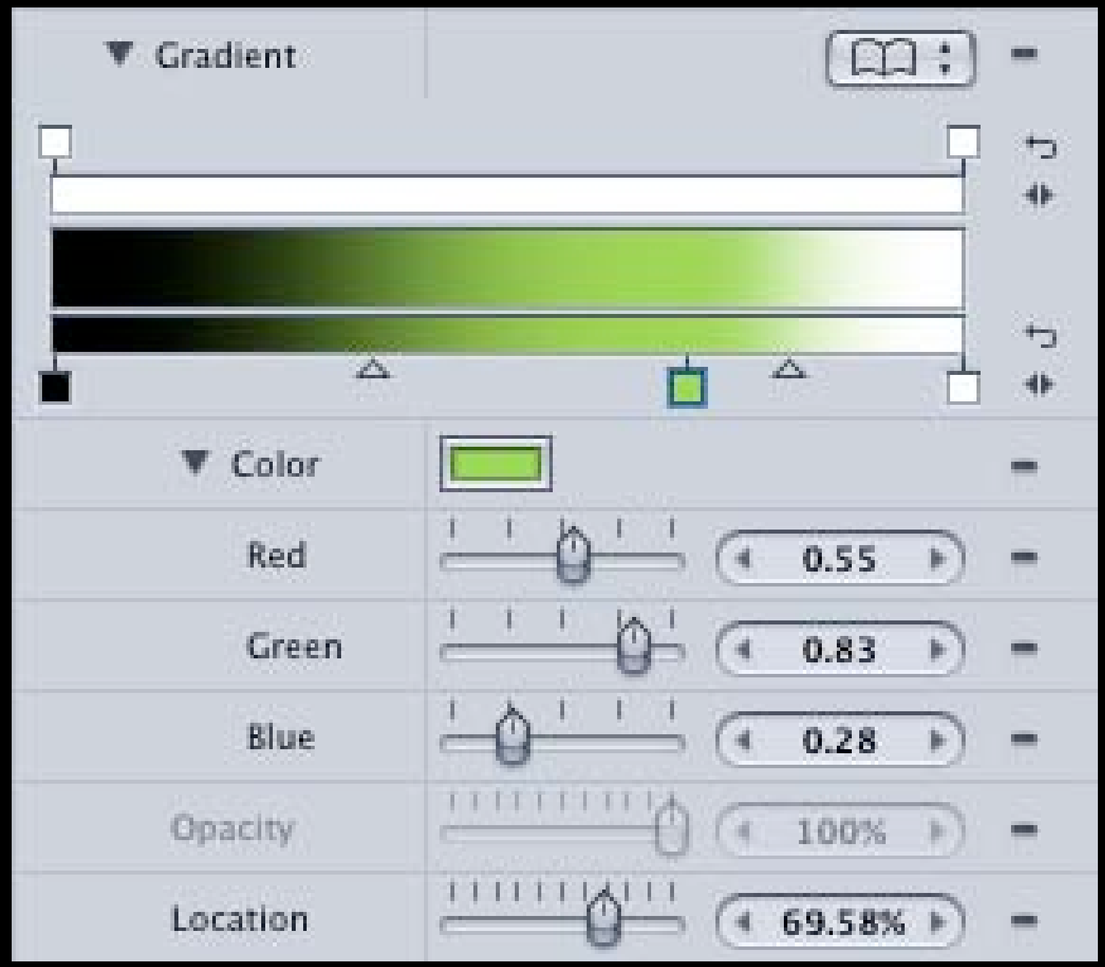
Histogram Value Setting

```
id setter = [_apiManager apiForProtocol:
                  @protocol(FxOptionalParameterSettingAPI)];

[setter setHistogramBlackIn:blackIn
                   BlackOut:blackOut
                    WhiteIn:whiteIn
                   WhiteOut:whiteOut
                      Gamma:gamma
                 forChannel:channelNum
                   fromParm:kMyHistogramParamID
                     atTime:renderInfo.frame];
```

Special Parameter Types

Gradient



Special Parameter Types

Gradient Creation

```
id creator = [_apiManager apiForProtocol:
                    @protocol(FxOptionalParameterCreationAPI)];

[creator addGradientWithName:@“Gradient”
                       parmId:kMyGradientParamID
                   parmFlags:kFxParameterFlag_DEFAULT];
```

Special Parameter Types

Gradient Value Retrieval

```
id getter = [_apiManager apiForProtocol:
                    @protocol(FxOptionalParameterRetrievalAPI)];
double samples[300];

[getter getGradientSamples:samples
                numSamples:300
                     depth:4   // 4 bytes, for 32-bit floats
                  fromParm:kMyGradientParamID
                    atTime:renderInfo.frame];
```

Demo

Optional Parameters

Pete Warden

http://developer.apple.com/wwdc2005/

Custom Parameters

Dave Howell

http://developer.apple.com/wwdc2005/

Custom Parameter Types

Creation

```
MyDataClass *defaultData = [MyDataClass emptyData];

id creator = [_apiManager apiForProtocol:
                  @protocol(FxParameterCreationAPI)];

[creator addCustomParameterWithName:@“Custom Thing”
                            parmId:kMyDataParamID
                      defaultValue:defaultData
                         parmFlags:kFxParameterFlag_CUSTOM_UI |
                                   kFxParameterFlag_NOT_ANIMATABLE |
                                   kFxParameterFlag_DONT_DISPLAY_IN_DASHBOARD];
```

Custom Parameter Types

Implement two <NSCoding> methods

```
@implementation MyDataClass

- (void)encodeWithCoder:(NSCoder *)coder
{
   [coder encodeObject:_data forKey:@“Data”];
}

- (void)initWithCoder:(NSCoder *)decoder
{
   self = [super init];
   _privateData = [[decoder decodeObjectForKey:@“Data”] retain];
   return self;
}
//...
@end
```

Custom Parameter Types

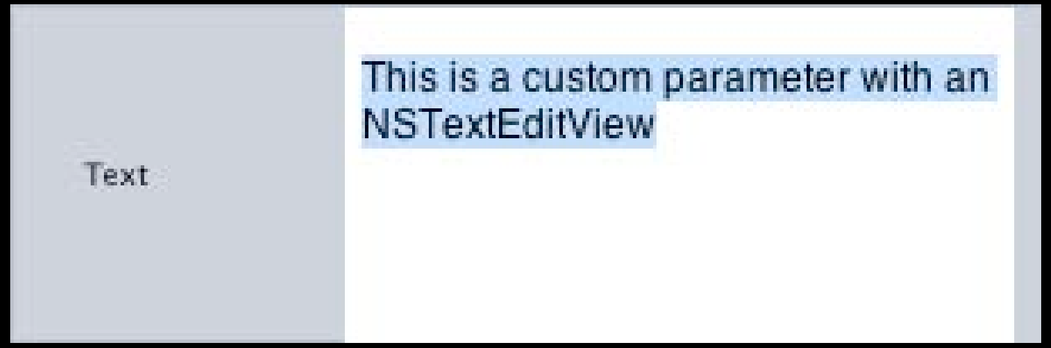
Restrictions

- No explicit support for keyframe animation
  - Create with `kFxParameterFlag_NOT_ANIMATABLE`
- `NSCoding` implementation must do keyed coding
  - Use `-encodeObject:forKey:` in `-encodeWithCoder:` method
  - Use `-decodeObjectForKey:` in `-initWithCoder:` method

Custom Parameter UI

Custom NSView in Inspector

- Subclass of `NSView`
  - May be designed in Interface Builder
- Works with standard or custom parameter types



Custom Parameter UI

Add FxCustomParameterViewHost protocol to Info.plist

```
<key>ProPlugPlugInList</key>
<array>
  <dict>
     ...
     <key>protocolNames</key>
     <array>
        <string>FxGenerator</string>
        <string>FxCustomerParameterActionAPI</string>
     </array>
     ...
  </dict>
  ...
</array>
```

Custom Parameter UI

Assigning an NSView to a parameter

- Create parameter with `kFxParameterFlag_CUSTOM_UI` flag
- Conform to <FxCustomParameterViewHost> protocol

```
- (NSView *)createViewForParm:(UInt32)parmId
{
  NSView *view = NULL;
  if ( parmId == kMyCustomItemID )
  {
     view = [[MyTextView alloc] initWithFrame:NSMakeRect( 0, 0, 200, 100 )
                                   andManager:_apiMgr
                                      forParm:parmId];
  }
  return view;
}
```

Custom Parameter UI

Implementing custom view

- In Motion, view appears in Inspector
- View may be resizable
- May be any subclass of NSView

Custom Parameter UI

Responding to events

- Like any other `NSView`
  - Override `NSView` and `NSResponder` methods
  - Assign contextual menu with `-setMenu:` or `-menuForEvent:`
  - Respond to events from key, mouse, scroll-wheel, tablet, *etc.*
- View's controller may change parameter value, state
  - To access parameters, must use `FxCustomParameterActionAPI`

Custom Parameter UI

FxCustomParameterActionAPI

- Prepares host app for parameter access

```
id actionAPI = [_apiMgr apiForProtocol:
                    @protocol(FxCustomParameterActionAPI)];
[actionAPI startAction:self];

// ... access parameters

[actionAPI endAction:self];
```

Custom Parameter UI

FxCustomParameterActionAPI

- Can also return current time

```
double time = [actionAPI currentTime];

[getAPI getFloatValue:&floatValue
            fromParm:kMySliderID
              atTime:time];
```

On-Screen Controls

Pete Warden

http://developer.apple.com/wwdc2005/

On-Screen Controls

What are they?

- Point parameters give built-in OSCs
- A way of drawing custom controls in the canvas
- For example, Kaleidoscope filter

On-Screen Controls

Drawing

- Drawn using OpenGL
- Composited over window
- Choice of drawing coordinates
  - Object
  - Window
  - Document
- Use antialiasing

On-Screen Controls

Events

- Usual mouse and keyboard events
- Set parameters based on input

On-Screen Controls

Selecting

- Draw controls using GL_SELECT
  - Just like normal drawing
  - No textures
  - No antialiasing
- Draw each part with a different ID

OpenGL Arcana

PBuffers

- What are they?
- What are they used for?

OpenGL Arcana

PBuffer Creation/Deletion

- Don’t do it every frame
- Create once, and cache
- Re-create when your size changes
  - Optimize to reduce times done

OpenGL Arcana

Using PBuffers

- Use as a texture
- Can’t write and read to the same PBuffer
- Double buffer, and ping-pong

OpenGL Arcana

Fragment Programs

- Pixel shaders
- Creation/Deletion
- Using them
- Alternative languages

OpenGL Arcana

Floating Point Pixels

- Only on Tiger
- Use renderInfo depth to create PBuffers
- No blending or bilinear filtering
- Otherwise, all conversion is handled for you

OpenGL Arcana

Depth or Z Buffer

- None by default
- Must create PBuffer with one attached
- May hit blocky intersection issues

OpenGL Arcana

Size limits

- Current
  - 2K x 2K on all ATI
  - 4K x 4K on nVidia 128MB @ 8-bit
- Smaller, depends on VRAM, for float
- We clamp sizes internally
  - Input and output always legal
- If you create larger PBuffers than input or output, must check

More Information

http://developer.apple.com/wwdc2005/

Documentation, Sample Code, and Other Resources

http://developer.apple.com/wwdc2005/

Related Sessions

FxPlug Development	Pro Audio and Video Connection Friday 12:45 PM

Whom to Contact

Patrick Collins
Technology Manager
patrick@apple.com

Pro App SDK Group

proappsdk@group.apple.com

Q&A

Patrick Collins
Technology Manager
patrick@apple.com

http://developer.apple.com/wwdc2005/

WWDC2005

Worldwide Developers Conference